LÝ THUYẾT SỐ NGUYÊN TỐ, HỢP SỐ

Tài liệu tóm tắt khái niệm, cách phân biệt, thuật toán cơ bản và ứng dụng trong lập trình thi học sinh giỏi

1. Các khái niệm cơ bản

1.1. Số nguyên tố

Số nguyên tố là số tự nhiên lớn hơn 1 và chỉ có đúng hai ước dương là 1 và chính nó.

Ví dụ: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29 là các số nguyên tố.

> *Chú ý: 2 là số nguyên tố chẵn duy nhất. Mọi số nguyên tố lớn hơn 2 đều là số lẻ.*

1.2. Hợp số

Hợp số là số tự nhiên lớn hơn 1 và có nhiều hơn hai ước dương. Nói cách khác, hợp số có thể phân tích thành tích của hai số tự nhiên nhỏ hơn nó.

Ví dụ: 4, 6, 8, 9, 10, 12, 15 là hợp số.

Ta có: 4 = 2 x 2, 6 = 2 x 3, 9 = 3 x 3, 12 = 3 x 4.

1.3. Số không là số nguyên tố và cũng không là hợp số

Các số 0 và 1 không được xem là số nguyên tố, cũng không được xem là hợp số.

Lý do:

- 0 có vô số ước và không phù hợp với định nghĩa số nguyên tố/hợp số trong số tự nhiên dương.
- 1 chỉ có đúng một ước dương là 1, nên không đủ hai ước để là số nguyên tố và cũng không có nhiều hơn hai ước để là hợp số.

> *Kết luận quan trọng: Trong chương trình, khi kiểm tra số nguyên tố, luôn xử lý riêng trường hợp n < 2.*

2. Bảng phân biệt nhanh

Loại số	Điều kiện	Ví dụ
Số nguyên tố	Lớn hơn 1, có đúng 2 ước dương: 1 và chính nó	2, 3, 5, 7, 11, 13
Hợp số	Lớn hơn 1, có nhiều hơn 2 ước dương	4, 6, 8, 9, 10, 12
Không là nguyên tố, không là hợp số	Không thỏa hai định nghĩa trên	0, 1

3. Cách kiểm tra một số có phải số nguyên tố

3.1. Ý tưởng thử chia

Để kiểm tra số n có phải số nguyên tố hay không, ta thử xem n có chia hết cho số nào từ 2 đến n - 1 không. Nếu có thì n là hợp số; nếu không thì n là số nguyên tố.

Tuy nhiên cách này chậm khi n lớn. Ta có thể tối ưu bằng cách chỉ kiểm tra đến căn bậc hai của n.

3.2. Vì sao chỉ cần kiểm tra đến n?

Nếu n là hợp số thì n = a x b, với 1 < a b < n. Khi đó a n. Vì vậy nếu không tìm thấy ước nào từ 2 đến n thì n không thể là hợp số.

3.3. Thuật toán

1. Nếu n < 2 thì n không phải số nguyên tố.
2. Duyệt i từ 2 đến khi i x i n.
3. Nếu n chia hết cho i thì n không phải số nguyên tố.
4. Nếu duyệt xong không có ước nào thì n là số nguyên tố.

3.4. Code C++

```
#include <bits/stdc++.h>
using namespace std;

bool laSoNguyenTo(long long n) {
    if (n < 2) return false;
    for (long long i = 2; i * i <= n; i++) {
        if (n % i == 0) return false;
    }
    return true;
}

int main() {
    long long n;
    cin >> n;
    cout << (laSoNguyenTo(n) ? "YES" : "NO");
    return 0;
}
```

Độ phức tạp: O(n).

4. Sàng Eratosthenes

Khi cần tìm tất cả các số nguyên tố từ 1 đến N, ta nên dùng sàng Eratosthenes. Đây là thuật toán rất quan trọng trong các bài toán số học.

4.1. Ý tưởng

1. Ban đầu giả sử mọi số từ 2 đến N đều là số nguyên tố.
2. Bắt đầu từ số nguyên tố nhỏ nhất là 2.
3. Loại bỏ các bội của 2 lớn hơn 2: 4, 6, 8, ...
4. Tiếp tục với 3, loại bỏ 6, 9, 12, ...
5. Tiếp tục như vậy đến N.

4.2. Code C++

```
#include <bits/stdc++.h>
using namespace std;

const int MAXN = 1000000;
bool prime[MAXN + 1];

void sangNguyenTo(int n) {
    for (int i = 0; i <= n; i++) prime[i] = true;
    prime[0] = prime[1] = false;

    for (int i = 2; i * i <= n; i++) {
        if (prime[i]) {
            for (int j = i * i; j <= n; j += i) {
                prime[j] = false;
            }
        }
    }
}

int main() {
    int n;
    cin >> n;
    sangNguyenTo(n);

    for (int i = 2; i <= n; i++) {
        if (prime[i]) cout << i << ' ';
    }
    return 0;
}
```

Độ phức tạp thường dùng: O(N log log N), phù hợp với N lớn như 10^6 hoặc 10^7.

5. Phân tích thừa số nguyên tố

Mọi số nguyên dương lớn hơn 1 đều có thể phân tích duy nhất thành tích các số nguyên tố, nếu không xét thứ tự các thừa số.

Ví dụ: 60 = 2^2 x 3 x 5; 84 = 2^2 x 3 x 7.

Code C++ phân tích thừa số nguyên tố

```cpp
#include <bits/stdc++.h>
using namespace std;

int main() {
    long long n;
    cin >> n;

    for (long long i = 2; i * i <= n; i++) {
        int dem = 0;
        while (n % i == 0) {
            dem++;
            n /= i;
        }
        if (dem > 0) {
            cout << i << "^" << dem << "\n";
        }
    }

    if (n > 1) {
        cout << n << "^1\n";
    }
    return 0;
}
```

6. Ứng dụng của số nguyên tố

6.1. Tìm ước, số lượng ước, tổng các ước

Khi đã phân tích n = p1^a1 x p2^a2 x ... x pk^ak, ta có thể tính nhanh nhiều đại lượng liên quan đến ước của n.

Số lượng ước của n là: (a1 + 1)(a2 + 1)...(ak + 1).

Ví dụ: 60 = 2^2 x 3^1 x 5^1 nên số lượng ước là (2 + 1)(1 + 1)(1 + 1) = 12.

6.2. Tìm ƯCLN và BCNN

Nếu phân tích A và B thành thừa số nguyên tố, ta có thể tính ƯCLN và BCNN như sau:

- ƯCLN lấy các thừa số nguyên tố chung với số mũ nhỏ nhất.
- BCNN lấy tất cả thừa số nguyên tố xuất hiện với số mũ lớn nhất.

Ví dụ: A = 60 = 2^2 x 3 x 5, B = 84 = 2^2 x 3 x 7. Khi đó ƯCLN = 2^2 x 3 = 12, BCNN = 2^2 x 3 x 5 x 7 = 420.

6.3. Đếm số nguyên tố trong đoạn

Với nhiều truy vấn [L, R], ta có thể sàng trước đến N, sau đó tạo mảng cộng dồn: prefix[i] = số lượng số nguyên tố từ 1 đến i.

Kết quả cho đoạn [L, R] là prefix[R] - prefix[L - 1].

6.4. Mật mã và an toàn thông tin

Số nguyên tố lớn được dùng trong mật mã học, tiêu biểu là RSA. Ý tưởng quan trọng là nhân hai số nguyên tố lớn thì dễ, nhưng phân tích một số rất lớn thành tích hai số nguyên tố ban đầu là rất khó.

7. Các lỗi thường gặp khi học sinh lập trình

- Quên xử lý n < 2, làm cho 0 hoặc 1 bị nhận nhầm là số nguyên tố.
- Duyệt i <= n thay vì i * i <= n, khiến chương trình chậm.
- Dùng int cho i * i khi n lớn, có thể bị tràn số. Nên dùng long long.
- Trong sàng, quên đặt prime[0] = prime[1] = false.
- Bắt đầu loại bội từ 2 * i thay vì i * i. Cách này vẫn đúng nhưng chậm hơn.

8. Bài tập luyện tập

Bài	Nội dung	Gợi ý thuật toán
1	Kiểm tra n có phải số nguyên tố không.	Thử chia đến n
2	Liệt kê các số nguyên tố không vượt quá N.	Sàng Eratosthenes
3	Đếm số nguyên tố trong đoạn [L, R] với nhiều truy vấn.	Sàng + mảng cộng dồn
4	Phân tích n thành thừa số nguyên tố.	Thử chia đến n
5	Tính ƯCLN, BCNN bằng phân tích nguyên tố.	Lấy min/max số mũ

9. Tóm tắt cần nhớ

- Số nguyên tố: lớn hơn 1, có đúng hai ước dương.
- Hợp số: lớn hơn 1, có nhiều hơn hai ước dương.
- 0 và 1 không là số nguyên tố, cũng không là hợp số.
- Kiểm tra nguyên tố hiệu quả bằng cách thử chia đến n.
- Sàng Eratosthenes dùng để tìm nhiều số nguyên tố nhanh.
- Số nguyên tố có nhiều ứng dụng trong phân tích thừa số, ƯCLN, BCNN, đếm số nguyên tố và mật mã.